**Assignment 2 – k-nearest neighbours and k-fold cross validation**

In this lab, the goal is the implement kNN model and perform 5-fold cross validation to choose the best k-NN model.

The kNN model is achieved by calculating the Euclidean distance between the test point and all the other points, then picking out k numbers of nearest point, and find the average of the corresponding target values.

Cross validation method is built based on the kNN model where the training data set is split into 5 parts, taking 4 sets as new training set and 1 for test. The kNN model is run through all 5 combinations and the average training error for all combinations is calculated to be the cross-validation error.

Plots of training error and the cross-validation error for all k-NN models:

A graph with blue dots

Description automatically generated

Figure 1: Training error vs k

A graph with blue dots

Description automatically generated

Figure 2: Cross-Validation Error vs K

A graph of a graph

Description automatically generated with medium confidence

Underfitting

Overfitting

Optimal

Figure 3: Training Error & Cross-Validation Error vs K

Figure 3 shows the comparison between the training error and the validation error. In the lecture, the error trends should have been shown as figure 4. In comparison to figure 3, the training error follows the exact trend. For the cross-validation error, the trend is slightly off as the slope increases relatively more rapidly. However, the trend is still acceptable as the rising trend follows the expected values.

A graph with lines and dots

Description automatically generated

Figure 4: Lecture Example on Error Comparison

Identify k values for underfitting, overfitting, region of optimal capacity:

A graph with red and blue lines

Description automatically generatedA graph of a training data

Description automatically generated

Figure 5: K = 1, Overfitting Figure 6: k = 50, Underfitting

\* The “training data” label should be “test data”

Figure 3 also depicts the three regions of model fitting. The beginning of the converging trend between training error and cross-validation error can be identified as the overfitting region. This region can be roughly identified from . By picking , and plot the prediction model with 1NN prediction model, figure 5 shows the overfitting of the model. Additionally, the overfitting of the model can be depicted by the “fuzzy” trend in the prediction model as it overcompensates for every single test points. Figure 6 is a great example of underfitting, with , locating in the underfitting region in figure 3. The prediction model in figure 6 failed to capture any trend that the test set possesses as it does not have the capacity to do such. The underfitting region can be roughly identified around . The starting point of the underfitting region is not quite accurate as there is no significant change or threshold in the uprising trend.

A graph of a training data

Description automatically generated

Figure 7: k = 20, Optimal Capacity

\* The “training data” label should be “test data”

The optimal capacity is located between from figure 3. Figure 7 is plotted with kNN where . It is identifiable that the prediction model is able to capture the general trend presented by the test set and it sticks quite close to the true function.

|  |  |  |
| --- | --- | --- |
| k | Training Error | Cross-Validation Error |
| 1 | 0 | 0.388736637 |
| 2 | 0.212587084 | 0.324139225 |
| 3 | 0.243156848 | 0.321459716 |
| 4 | 0.263938216 | 0.303941702 |
| 5 | 0.269442196 | 0.310970292 |
| 6 | 0.272885461 | 0.301482272 |
| 7 | 0.269586102 | 0.30332941 |
| 8 | 0.274867613 | 0.331910193 |
| 9 | 0.271726328 | 0.345730078 |
| 10 | 0.279589701 | 0.362735111 |
| 11 | 0.280224343 | 0.359301325 |
| 12 | 0.286453203 | 0.380431578 |
| 13 | 0.287561928 | 0.376012238 |
| 14 | 0.294150336 | 0.391959891 |
| 15 | 0.292626507 | 0.399290667 |
| 16 | 0.298214347 | 0.421636596 |
| 17 | 0.300569438 | 0.428784879 |
| 18 | 0.309027219 | 0.44578767 |
| 19 | 0.309244678 | 0.469197156 |
| 20 | 0.317489012 | 0.491872144 |
| 21 | 0.319286122 | 0.516287381 |
| 22 | 0.32680958 | 0.545398063 |
| 23 | 0.326751893 | 0.558520575 |
| 24 | 0.334186472 | 0.585085422 |
| 25 | 0.335211958 | 0.604994863 |
| 26 | 0.342738372 | 0.635014805 |
| 27 | 0.343629364 | 0.654644681 |
| 28 | 0.348789961 | 0.673450132 |
| 29 | 0.349648223 | 0.696230791 |
| 30 | 0.354195428 | 0.71484126 |
| 31 | 0.356552942 | 0.732246283 |
| 32 | 0.364537132 | 0.753871932 |
| 33 | 0.365488549 | 0.784624546 |
| 34 | 0.369743328 | 0.807725472 |
| 35 | 0.373260483 | 0.821985559 |
| 36 | 0.37789315 | 0.847421557 |
| 37 | 0.379414711 | 0.86914684 |
| 38 | 0.383887751 | 0.895904394 |
| 39 | 0.384325359 | 0.914878022 |
| 40 | 0.388833187 | 0.942227409 |
| 41 | 0.390495433 | 0.967460322 |
| 42 | 0.394451649 | 0.989802041 |
| 43 | 0.395503371 | 1.009647557 |
| 44 | 0.399322071 | 1.028759883 |
| 45 | 0.402128611 | 1.043554324 |
| 46 | 0.405670228 | 1.064278027 |
| 47 | 0.40852264 | 1.075445398 |
| 48 | 0.411718286 | 1.081461518 |
| 49 | 0.415231129 | 1.092434795 |
| 50 | 0.419673354 | 1.105681752 |
| 51 | 0.42412163 | 1.109491771 |
| 52 | 0.428321573 | 1.111090721 |
| 53 | 0.433472674 | 1.115476079 |
| 54 | 0.438622837 | 1.118114367 |
| 55 | 0.444037507 | 1.119935637 |
| 56 | 0.450222341 | 1.120223897 |
| 57 | 0.455542047 | 1.12122262 |
| 58 | 0.462981145 | 1.116624899 |
| 59 | 0.4689268 | 1.122493313 |
| 60 | 0.476220652 | 1.126610178 |

Table 1: k vs training error vs cross-validation error

Best model:

A graph with blue dots and red lines

Description automatically generated

Figure 6: Best kNN model prediction - Test Set Performance

\* The “training data” label should be “test data”

The best k-value is determined by locating the minimum cross-validation error in the training set. Looking at table 1 and using the argmin() function, it is determined that the best kNN prediction model occurs when This k hyperparameter value is located correctly in the optimal capacity. The test error (RMSE) is calculated to be 0.30087546405653753.

A graph of a test set performance measurement

Description automatically generated

Figure 7: Assignment 1 - Optimal Model performance in test data

Figure 9 shows the best prediction model from of assignment 1 where a least squares linear regression formula was used. The best prediction function is written as above. The prediction model has a test error (RMSE) of 0.3949494709298686.

|  |  |
| --- | --- |
| kNN, k = 6 | Least squares linear regression |
| 0.30087546405653753 | 0.3949494709298686 |

By comparing the test errors from both model, it is obvious that kNN with k = 6 has a lower test error. In terms of actual fitting, figure 9 (least squares) definitely has a smoother fitting than figure 8 (kNN). However, the kNN fitting in figure 8 shows a closer fit to the true function than least squares model. Therefore, it is confident to say that in this case, the kNN model with k = 6 is a better prediction model than the least squares linear regression mode.